Entender o programa First Responder (Secure Firewall Edition)

Contents

Introduction
Prerequisites
Requirements
Componentes Utilizados
Informações de Apoio
E-mail automatizado
Script / Comandos
Motivo para este e-mail
E-mail automatizado
Bloco de introdução
Bloco de Solicitação de Dados
Comando gerado
Script Firepower.py
Automação
Interativo
Saída esperada do script
Problemas comuns
Segurança de e-mail/Regravação de URL
Etapas a serem resolvidas
Falha de DNS
Etapas a serem resolvidas
Falha ao abrir/criar arquivo de log
Etapas a serem resolvidas
Falha ao abrir/gravar arquivo de notificação
Etapas a serem resolvidas
Falha ao Bloquear o Arquivo sf_troubleshoot.pid
Etapas a serem resolvidas
Problemas de upload
Etapas a serem resolvidas

Introduction

Este documento descreve o uso e a implementação do programa First Responder para o Cisco Secure Firewall.

Prerequisites

Requirements

Não existem requisitos específicos para este documento.

Componentes Utilizados

Este documento é baseado nos produtos Cisco Secure Firewall.

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. Se a rede estiver ativa, certifique-se de que você entenda o impacto potencial de qualquer comando.

Informações de Apoio

O programa First Responder foi criado pelo TAC para facilitar e agilizar o fornecimento de dados de diagnóstico para os casos abertos. Há dois componentes principais que compõem o programa:

E-mail automatizado

Este e-mail é enviado no início do caso com instruções sobre como coletar e carregar dados de diagnóstico para análise do TAC. Há várias tecnologias que aproveitam esse sistema, e cada e-mail é mapeado para a "Tecnologia" e a "Subtecnologia" escolhidas quando o caso é criado.

Script / Comandos

Cada implementação do programa First Responder tem sua própria maneira exclusiva de lidar com a coleta e a entrega de dados. A implementação do Secure Firewall utiliza o script firepower.py Python criado pelo TAC para realizar isso. O processo de e-mail automatizado gera um comando de uma linha, exclusivo para esse caso específico, que pode ser copiado e colado na CLI dos dispositivos do Secure Firewall para execução.

Motivo para este e-mail

Há determinadas tecnologias que são ativadas para o programa de socorristas. Isso significa que sempre que um caso é aberto contra uma dessas tecnologias habilitadas, um e-mail de socorrista é enviado. Se você receber um e-mail de resposta e não acreditar que as solicitações de dados são relevantes, não hesite em ignorar a comunicação.

Para o caso de uso do Secure Firewall, o programa de socorristas é limitado ao software Firepower Threat Defense (FTD). Se você executar uma base de código do Adaptive Security Appliance (ASA), ignore este e-mail. Como esses dois produtos são executados no mesmo hardware, geralmente se observa que os casos de ASA são criados no espaço de tecnologia do Secure Firewall, que gera o e-mail do respondente principal.

E-mail automatizado

Aqui está um exemplo de e-mail automatizado enviado como parte deste programa:

```
From: first-responder@cisco.com <first-responder@cisco.com>
Sent: Thursday, September 1, 2022 12:11 PM
```

```
To: John Doe <john.doe@cisco.com>
Cc: attach@cisco.com
Subject: SR 666666666 - First Responder Automated E-mail

Dear John,

In an effort to resolve your case faster it may be necessary to collect some diagnostic data
from your environment.
Based on the problem statement you provided, below are a few pieces of data that would help
speed the resolution and the steps to collect them:

*** Troubleshoot File ***

* Connect to the device using SSH
* Issue the command expert, skip this step for FMC version 6.4.x and earlier
* Issue the command sudo su
* When prompted for the password, enter your password.
* For FMC 6.4 or FTD 6.7 and later issue the command
curl -s -S https://cxd.cisco.com/public/ctfr/firepower.py | python - -c 666666666 -t
aBcDeFgHiJkLmNoP --auto-upload &

* For FMC 6.3 or FTD 6.6 and earlier issue the command
curl -k -s -S https://cxd.cisco.com/public/ctfr/firepower.py | python - -c 666666666 -t
aBcDeFgHiJkLmNoP --auto-upload &

For more information on what this command does, or to understand why you are receiving this e-
mail - please refer to
<LINK_TO_THIS_ARTICLE>

For 6.3 and earlier versions we recommend confirming cxd.cisco.com resolves to <CURRENT_CXD_IP1>
or
<CURRENT_CXD_IP2>. Furthermore, we recommend validating the SHA checksum of the file by running
url -s -k https://cxd.cisco.com/public/ctfr/firepower.py | shasum which should output
<CURRENT_SHA>.

If you are unable to upload troubleshooting files (or would prefer not to), please let us know
what
hardware and software version ou are running if you have not already.

Sincerely, First Responder Team
```

Os e-mails automatizados do programa de socorristas são divididos em duas partes, conhecidas como o bloco de introdução e o bloco de solicitação de dados.

Bloco de introdução

O bloco de introdução é uma sequência de caracteres estática que é incluída em cada e-mail de resposta inicial. Esta frase introdutória serve simplesmente para fornecer contexto ao(s) bloco(s) de solicitação de dados. Aqui está um exemplo de um bloco de introdução:

```
Dear <NAME>,

In an effort to resolve your case faster it may be necessary to collect some diagnostic data
from your environment.
Based on the problem statement you provided, below are a few pieces of data that would help
speed the resolution
and the steps to collect them:
```

Bloco de Solicitação de Dados

Os blocos de solicitação de dados são o coração do programa de primeiros respondentes. Cada

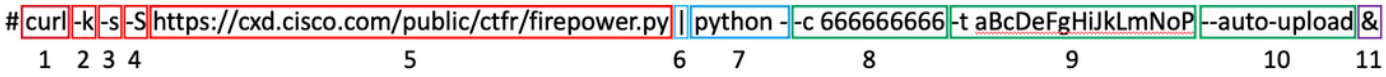
bloco é um conjunto predefinido de etapas para coletar dados para uma determinada tecnologia. Conforme mencionado na seção **Informações de Apoio**, cada bloco de solicitação de dados é mapeado para uma tecnologia específica. Essa é a mesma tecnologia escolhida para abrir um caso de suporte. Normalmente, o e-mail automatizado contém um único bloco de solicitação de dados. No entanto, se a tecnologia escolhida tiver mais de um bloco de solicitação de dados mapeado para ela, várias solicitações de dados serão incluídas no e-mail. Aqui está um formato de exemplo do bloco de Solicitação de Dados com várias solicitações de dados:

```
*** <REQUEST NAME 1> ***

<REQUEST 1 STEPS>

*** <REQUEST NAME 2> ***

<REQUEST 2 STEPS>
```

Por exemplo, no caso do Firewall seguro, vários blocos de solicitação de dados são frequentemente incluídos quando uma solicitação é feita para assistência com problemas de VPN de acesso remoto (RA-VPN) com Defesa contra ameaças (FTD) Firepower, já que a tecnologia VPN também tem um bloco de solicitação de dados mapeado configurado para assistência na coleta dos pacotes DART.

Comando gerado

Especificamente para o caso de uso do Secure Firewall, um comando exclusivo de uma linha é gerado para cada caso como parte do e-mail automatizado. Aqui está uma divisão da estrutura do comando de uma linha:



1. O comando curl é usado para baixar a versão mais recente do script firepower.py
2. O sinalizador **-k** é uma opção para que o curl ignore erros de certificado durante a conexão.
3. O sinalizador **-s** é uma opção para que o curl seja executado no modo silencioso. Isso é usado para suprimir a saída de curva normal, pois é ruidosa.
4. O sinalizador **-S** é uma opção para que o curl mostre erros. Isso é usado para forçar o curl a ainda mostrar erros de saída mesmo com a opção silent ativada.
5. A URL onde a versão mais recente do script firepower.py está hospedada. Esse caminho instrui o comando curl a extrair o corpo mais recente do script a ser executado.
6. Este é um pipe Linux, que passa a saída do comando curl (o conteúdo de um script python) para uma instrução de execução na próxima etapa.
7. Neste passo, o binário python no dispositivo é chamado com um "-" adicional. Isso instrui python que a origem é obtida do stdin (já que o conteúdo do script é transportado do curl).
8. O sinalizador **-c** é um argumento de entrada para o script firepower.py, que indica o número do caso para o qual os dados devem ser carregados. O valor 666666666 após essa opção é o número do caso de exemplo.
9. O sinalizador **-t** é um argumento de entrada para o script firepower.py, que indica um token exclusivo (senha) que foi gerado para esse caso específico. O valor aBcDeFgHiJkLmNoP após esta opção é o token de exemplo para este caso.
10. O sinalizador **—autoupload** é um argumento especial para o script firepower.py, que indica que o script deve ser executado no modo de automação. Mais informações sobre isso

podem ser encontradas na seção específica do script.

11. O comando **&** instrui todo este comando para ser executado em segundo plano, o que permite que o usuário continue a interagir com seu shell enquanto o script é executado.

Note: A bandeira -k é necessária para qualquer versão do FMC anterior à 6.4 e qualquer versão do FTD anterior à 6.7, uma vez que o certificado de raiz utilizado pelo CXD não era de confiança dos dispositivos Firepower até o FMC versão 6.4 e o FTD versão 6.7, o que provoca uma falha na verificação do certificado.

Script Firepower.py

O objetivo principal do script é gerar e carregar um pacote de diagnóstico do dispositivo Secure Firewall chamado de "solução de problemas". Para gerar esse arquivo de solução de problemas, o script firepower.py simplesmente chama o script sf_troubleshoot.pl interno responsável pela criação desse pacote. Esse é o mesmo script chamado quando geramos uma solução de problemas a partir da GUI. Além do arquivo de solução de problemas, o script também pode coletar outros dados de diagnóstico que não estão incluídos como parte do pacote de solução de problemas. Atualmente, os únicos dados adicionais que podem ser coletados são os arquivos centrais - mas isso pode ser expandido no futuro se houver necessidade. O script pode ser executado no modo "Automação" ou "Interativo":

Automação

Esse modo é ativado quando usamos a opção "—autoupload" quando executamos o script. Esta opção desativa os prompts interativos, ativa a coleta de arquivos principais e carrega automaticamente os dados no caso. O comando de uma linha gerado pelo e-mail automático inclui a opção "—autouload".

Interativo

Este é o modo de execução padrão do script. Nesse modo, o usuário recebe avisos para confirmar se deve ou não coletar dados de diagnóstico adicionais, como arquivos de núcleo. Independentemente do modo de execução, uma saída significativa é impressa na tela e registrada em um arquivo de registro para indicar o andamento da execução dos scripts. O script em si é amplamente documentado através de comentários de código em linha e pode ser descarregado / revisado em https://cxd.cisco.com/public/ctfr/firepower.py.

Saída esperada do script

Aqui está um exemplo de uma execução bem-sucedida do script:

```
root@ftd:/home/admin# curl -k -s -S https://cxd.cisco.com/public/ctfr/firepower.py | python - -c
666666666 -t aBcDeFgHiJkLmNoP --auto-upload &
[1] 26422
root@ftd:/home/admin#
`/var/common/first_responder_notify` successfully uploaded to 666666666
Running sf_troubleshoot.pl command to create a troubleshoot file...
Troubleshoot file successfully generated at /ngfw/var/common/results-08-30-2022--135014.tar.gz
Attempting to upload troubleshoot to case...
##################################################################################################
############################ 100.0%
```

```
`/ngfw/var/common/results-08-30-2022--135014.tar.gz` successfully uploaded to 666666666
Found the following core files:
(0 B) - /ngfw/var/common/core_FAKE1.gz
(0 B) - /ngfw/var/common/core_FAKE2.gz
(0 B) - /ngfw/var/common/core_FAKE3.gz
Successfully created /ngfw/var/common/cores_666666666-1661867858.tar.gz
Attempting core file upload...
################################################################################################
############################## 100.0%
`/ngfw/var/common/cores_6666666660-1661867858.tar.gz` successfully uploaded to 666666666
FINISHED!
```

Observe que este exemplo de saída inclui carregamentos de arquivos principais. Se não houver arquivos principais no dispositivo, uma mensagem "No core files found. Skipping core file processing" é apresentado.

Problemas comuns

Aqui estão alguns problemas comuns que você pode experimentar (em ordem de processo / execução):

Segurança de e-mail/Regravação de URL

Muitas vezes, observa-se que o usuário final tem algum nível de segurança de e-mail que regrava o URL. Isso altera o comando de uma linha gerado como parte do e-mail automatizado. Isso resulta em uma falha de execução, pois a URL para receber o script foi regravada e é inválida. Aqui está um exemplo do comando esperado de uma linha:

```
curl -s -S https://cxd.cisco.com/public/ctfr/firepower.py | python - -c 666666666 -t
aBcDeFgHiJkLmNoP --auto-upload &
```

Etapas a serem resolvidas

Se o URL no comando do e-mail for qualquer outro que não "https://cxd.cisco.com/public/ctfr/firepower.py", o URL provavelmente foi regravado em trânsito. Para corrigir esse problema, basta substituir o URL antes de executarmos o comando.

Falha de DNS

Este erro de curva geralmente é visto quando o dispositivo não consegue resolver a URL para baixar o script:

```
curl: (6) Could not resolve host: cxd.cisco.com
```

Etapas a serem resolvidas

Para corrigir esse problema, verifique as configurações DNS no dispositivo para garantir que ele possa resolver a URL corretamente para continuar.

Falha ao abrir/criar arquivo de log

Uma das primeiras coisas que o script tenta fazer é criar (ou abrir, se já existir) um arquivo de log

chamado first-responder.log no diretório de trabalho atual. Se essa operação falhar, um erro que indica um problema de permissão simples será exibido:

```
Permission denied while trying to create log file. Are you running this as root?
```

Como parte dessa operação, todos os outros erros são identificados e impressos na tela neste formato:

```
Something unexpected happened while trying to create the log file. Here is the error:
----------
```

```
----------
```

Etapas a serem resolvidas

Para corrigir esse erro, basta executar o script como um usuário administrativo, como "admin" ou "root".

Falha ao abrir/gravar arquivo de notificação

Como parte da execução do script, um arquivo de 0 byte chamado "first_responder_notify" é criado no sistema. Esse arquivo é carregado no caso como parte da automação desse programa. Esse arquivo é gravado no diretório "/var/common". Se o usuário que executa o script não tiver permissões suficientes para gravar arquivos neste diretório, o script exibirá o erro:

```
Failed to create file -> `/var/common/first_responder_notify`. Permission denied. Are you
running as root?
```

Etapas a serem resolvidas

Para corrigir esse erro, basta executar o script como um usuário administrativo, como "admin" ou "root".

> **Note**: Se um erro relacionado a não permissões for encontrado, um erro catch-all será impresso na tela "Unexpected error while trying to open file -> `/var/common/first_responder_notify`. Please check first-responder.log file for full error". O corpo completo da exceção pode ser encontrado em first-respondedor.log .

Falha ao Bloquear o Arquivo sf_troubleshoot.pid

Para garantir que apenas um processo de geração de solução de problemas seja executado de cada vez, o script de geração de solução de problemas tenta bloquear o arquivo /var/sf/run/sf_troubleshoot.pid antes de continuar. Se o script falhar ao bloquear o arquivo, um erro será exibido:

```
Failed to run the `sf_troubleshoot.pl` command - existing sf_troubleshoot process detected.
```

```
Please wait for existing process to complete.
```

Etapas a serem resolvidas

Na maioria das vezes, esse erro significa que uma tarefa separada de geração de solução de problemas já está em andamento. Às vezes, isso é resultado de usuários que executam acidentalmente o comando de uma linha duas vezes seguidas. Para corrigir esse problema, aguarde a conclusão do trabalho de geração de solução de problemas atual e tente novamente mais tarde.

> **Note**: Se ocorrer um erro no script sf_troubleshoot.pl, esse erro será exibido na tela "Unexpected PROCESS error while trying to run `sf_troubleshoot.pl` command. Please check first-responder.log file for full error". O corpo completo da exceção pode ser encontrado em first-respondedor.log .

Problemas de upload

Há uma função de carregamento comum no script que é responsável por todos os carregamentos de arquivos durante a execução dos scripts. Esta função é simplesmente um wrapper python para executar um comando curl upload para enviar os arquivos para o caso. Por isso, qualquer erro encontrado durante a execução retorna como um código de erro curl. No caso de uma falha de carregamento, este erro é exibido na tela:

```
[FAILURE] Failed to upload `/var/common/first_responder_notify` to 666666666. Please check the first-responder.log file for the full error
```

Verifique o arquivo first-responder.log para ver o erro completo. Normalmente, o arquivo first-responder.log se parece com:

```
08/29/2022 06:51:57 PM - WARNING - Upload Failed with the following error:
----------
Command '['curl', '-k', '--progress-bar',
'https://666666666:aBcDeFgHiJkLmNoP@cxd.cisco.com/home/',
'--upload-file', '/var/common/first_responder_notify']' returned non-zero exit status 6
----------
```

Etapas a serem resolvidas

Nesse caso, o curl retornou um status de saída de 6 , o que significa "Não foi possível resolver o host ". Esta é uma falha simples de DNS enquanto tentamos resolver o nome de host cxd.cisco.com . Consulte a documentação da curva para decodificar qualquer status de saída desconhecido.

Sobre esta tradução
A Cisco traduziu este documento com a ajuda de tecnologias de tradução automática e humana para oferecer conteúdo de suporte aos seus usuários no seu próprio idioma, independentemente da localização.

Observe que mesmo a melhor tradução automática não será tão precisa quanto as realizadas por um tradutor profissional.

A Cisco Systems, Inc. não se responsabiliza pela precisão destas traduções e recomenda que o documento original em inglês (link fornecido) seja sempre consultado.